MM804 Project Report: 3D Custom Cropping Tool

Shang Lu
1397476
slu2@ualberta.ca

with Dinghao Liu, Liwen Chen

***Abstract*—3D medical data such as CT usually need to be cropped to eliminate peripheral distractions. Based on Visualization ToolKit (VTK), we build a cropping tool to make the cropping process as user-custom as possible. In our tool, the area of interest can be selected by mouse click, while the size and shape of area can be adjusted via sliders. Data choosing panel and Reset function are even built in our tool enabling choosing data and re-cropping very convenient.**

***Index Terms*—Medical images, VTK, Data cropping**

I. Introduction

Medical images, such as computed tomography (CT) data, are very important for surgical planning. The original CT data usually cover a large volume of body including abundant information while the surgeons care only part of the data. Since the peripheral data can act as distractions, a cropping technique is needed to assist the surgeons in focusing on area of interest.

Some researhcers have already provided a segmentation technique helping medical image analysis [1]. However, cropping is different from segmentation. Segmentation is usually organ-based, still consisting peripheral image. Cropping on the contrary is based on the interest of user, eliminating useless data. That is why we are still interested in cropping tool.

The visualization toolkit (VTK) is very popular in medical data processing [2]. It supports many powerful features of visualization and computer graphic, including 3-dimension cropping technique [2], [3]. We believe we can build a cropping tool based on VTK. To make the tool powerful enough, we are willing to construct it with a user custom style. That is the users can conveniently set the size and location of the cropping area as they wish. This process requires high-level understanding of VTK. During research, we found that [4] and [3] provide some useful examples on how to deal with CT data and how to crop the data. However, these examples have large room for improvements.

This report is organized as follow. Section II introduces the method we used and implementation details in the project. Section III shows the sample results. Then we discuss the results and the project briefly in section IV. Finally a conclusion is shown in section V.

II. Methods

A. Basic idea

[4] gives us a very good example on how a quadric function controlled by sliders crops the CT data. However, we think a box-shape area is more appropriate to provide the feeling of cropping to the users. Also, it is indirect for moving a cropping area by slider. In our imagination, users should use their mouse to point out the region of interest in the data directly. We built our solution based on such idea. Additionally, the example on Github [4] is quite simple, lacking enough user interfaces. We will build up some user interface for choosing data and reset the cropping.

B. Module used

TKinter is the standard Python interface to the Tk GUI toolkit, available on most platform [5]. We use this module to build a start user panel for choosing data. A TKinter Entry is built to get the users' input of folder path, see Fig 2.

vtkDICOMImageReader reads DICOM images. DICOM stands for Digital Imaging in COmmunications and Medicine, which is a medical image file format widely used [3]. We need this module to read CT data.

vtkSuperquadric computes the implicit function and function gradient for a superquadric [3]. We use this module to define the cropping area, with SetCenter function to set the location, SetScale to set the size and SetPhiRoundness/SetThetaRoundness to set the shape.

vtkPicker shoots a ray into a graphic window and select the actor's bounding box which is intersected [3]. We build a **vtkPicker** so that users can select a point in the data through mouse click on the screen. Through Pick function, a 2D pixel location on the screen is transfered into 3D location in the data.

vtkSphereSource creates a sphere of specified radius centered at the origin [3]. We use this module to show the point selected by users. The radius of such sphere is set to be tiny so that it does not affect the view of cropping.

vtkImplicitFunctionToImageStencil converts our **vtkSuperquadric** function into a stencil. Then **vtkImageStencil** will use the stencil to combine images together [3].

vtkColorTransferFunction defines a transfer function for mapping a property to an RGB color value, while **vtkPiecewiseFunction** defines a piecewise function mapping [3]. We set both functions with parameters in previous assignment.

vtkVolume is used to represent a volumetric entity in a rendering scene, while **vtkVolumeProperty** represents common properties associated with volume rendering and **vtkSmartVolumeMapper** delegates to a specific volume map-

per based on rendering parameters and available hardware [3]. We use **vtkVolume** to render the data, with its **vtkSmartVolumeMapper** connected to the **vtkImageStencil** and its **vtkVolumeProperty** taking the **vtkColorTransferFunction** as color function and **vtkPiecewiseFunction** as scaler opacity.

vtkRenderer controls the rendering process for objects, then draws the image into the window created by **vtkRenderWindow**. **vtkRenderWindowInteractor** provides interaction mechanism for control events in the **vtkRenderWindow** [3]. All these modules are essential display pipeline in VTK. We set the **vtkVolume** as the input in **vtkRenderer** then connect the display pipeline together.

vtkSliderWidget builds a slider which is moved along a 1D parameter space, thus modifying the value of the widget sets parameter on other objects. Its representation is provided by **vtkSliderRepresentation2D** [3]. We use these modules to setup sliders so users can set the parameters of cropping area. All the sliders are located at the left edge of screen so that the data are not blocked, see Fig 3 to 6 for examples.

vtkTextWidget provides support for interactively placing text on the 2D overlay plane [3]. We use this module to build the "Reset" button and the instruction text. Clicking the "Reset" text, users can go back to the initial status easily.

AddObserver function allows users to adjust callback function to any VTK object [3]. We add an observer to each slider so it can adjust the corresponding parameter. We add another observer to **vtkTextWidget** so it can perform reset function. An observer is also added to the **vtkRenderWindowInteractor** so users can select the point in the data through interactor.

Fig 1 shows the main data flow in VTK.

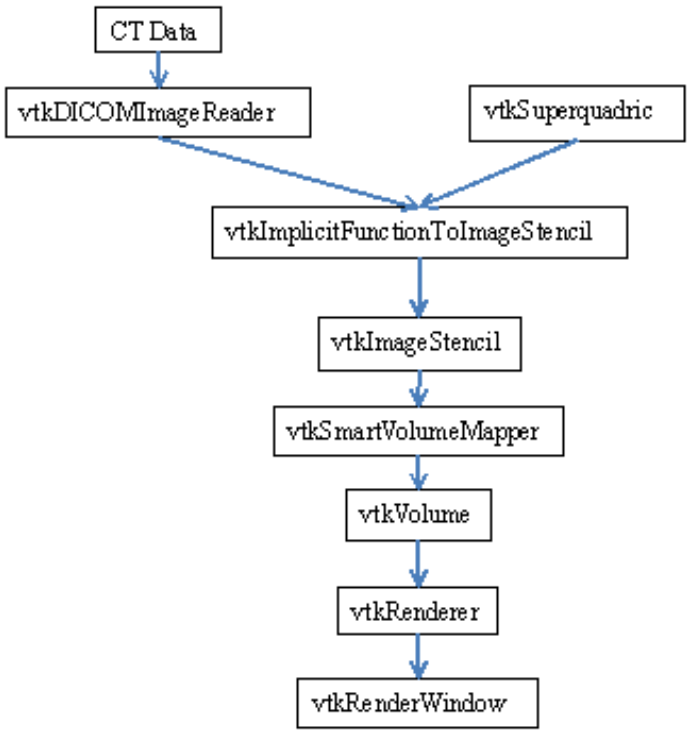

Fig. 1. Main data flow in VTK.

III. Results

In this project, python version 2.7, vtk version 7.1.1 are used.

A. Data choosing panel

When users begin to run the source code, a user-interface "Start" panel is displayed (see Fig 2). Users can input into the blank directly the directory path of the folder which contains the CT data. The path input should be full from the basic disk or the folder of the source code to the last level folder containing the DCM files. Finishing input, users can click the "Choose" button, then the directory is printed in the python console. At the same time the "Start" panel closes automatically and a VTK window is displayed.

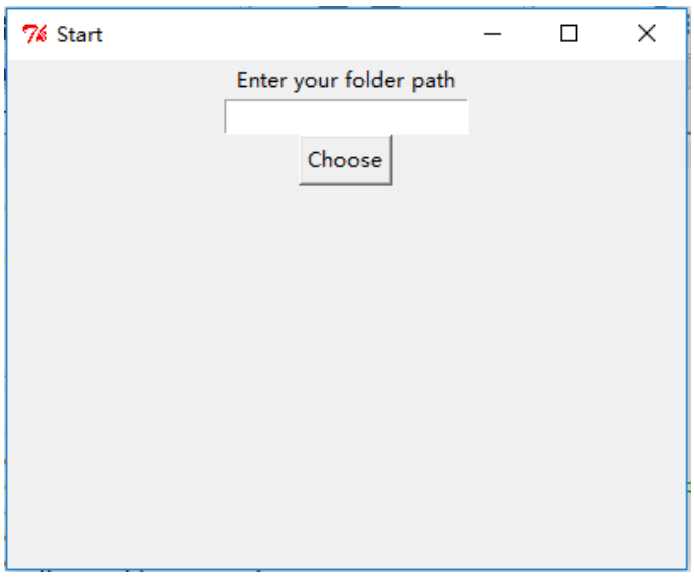

Fig. 2. Start panel.

B. Data cropping in VTK window

After users click "Choose" button on "Start" panel, the "Start" panel is closed and a VTK window is displayed, see Fig 3. The data are displayed in the center of the window. Users can interact with the data normally. In this project, the cropping area is defined by a superquadric function, originally centering at the center of the data with initial scale on each axis set to 500. So large amount of data are included in the cropping area. The PhiRoundness and ThetaRoundness of the superquadric function are 0 at the beginning.

Users can choose the point of their interest from the data, then right click the mouse at that point of interest. A white tiny sphere appears at the point selected indicating the center of the superquadric. The cropping area now centers the point selected with scale 100 on each axis, see Fig 4. Since the PhiRoundness and ThetaRoundness are still 0 at this time, the cropping area is cubic. On the left side of the window, 5 sliders are displayed,

controlling parameters of the superquadric. To adjust the shape of the cropping area, users can move the sliders with title "N/S Roundness" or "E/W Roundness". Moving such sliders from 0 to 1 will turn the cropping area from cube to sphere, see Fig 5 for example. To change the size of the cropping area, users can move the sliders named "Radius $\sim$". Moving such sliders change the scale on corresponding axis, see Fig 6.

Users can left-click the "Reset" text at the top-right corner to set the cropping area back to initial size, shape and location. So users can re-crop the data from beginning.

IV. Discussions

Through this project, we build a source code implementing custom cropping tool for 3D data. Starting with a user interaction panel, the project provides a convenient way to select the processing data. In cropping part, the center of area of interest can be selected directly by mouse click. This method really shortens the interaction procedure. Also several sliders are provided to adjust the parameters of the cropping area. We even build a Reset function, making the re-cropping more convenient.

At the beginning of the project, we proposed two solutions to the cropping tool based on the example we found [4]. One of them is a box shape cropping tool totally controlled by sliders, the other is a superquadric whose center is selected via mouse click. Both of them were demonstrated during presentation. Nevertheless, when we researched deeper, we found that we can combine both solutions together, getting the result in this report. The combination makes the project better, providing convenient shape and size adjustment while keeping the direct location selection.

However, the project is still far from perfect, requiring several improvements in the future. First, cropping style might be improved in the future. In this project, users can select the center of area freely. In future possible style, users might select the bounds of cropping area freely. Though we do not have enough time to test this idea in this project, we believe it can be implemented through VTK. Second, a step-by-step cropping, which at each step takes the output of last step as input, might be required in the future. This style might be important in reality, but we do not have enough time to carry it out.

A. Personal contribution

Based on research result [4], I proposed the basic ideas in this project. I wrote the code about data choosing panel and Reset function myself. I revised and improved the other part of the source code, for example the "select_point" function.

V. Conclusion

We proposed a solution to custom cropping for 3D data. The tool was generated successfully, thus users can crop the data as they wish. Data choosing interface and Reset function were added to the project, providing better user experiences.

Acknowledgment

We would like to thank Professor Kumaradevan Punithakumar for providing excellent lectures this term. We also would like to thank Xuping Fang for his previous work and experience.

References

[1] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *3D Vision (3DV), 2016 Fourth International Conference on*. IEEE, 2016, pp. 565–571.

[2] W. Narkbuakaew, S. Sotthivirat, D. Gansawat, P. Yampri, K. Koonsanit, W. Areeprayolkij, W. Sinthupinyo, and S. Watcharabutsarakham, "3d surface reconstruction of large medical data using marching cubes in vtk," in *The 3rd International Symposium on Biomedical Engineering*, 2008, pp. 64–67.

[3] I. Kitware. (2018, Apr.) Documentation — vtk. Kitware, Inc. New York, USA. [Online]. Available: https://www.vtk.org/documentation/

[4] Eclipse001. (2018, Apr.) Croppingtoolwithvtk. GitHub, Inc. San Francisco, USA. [Online]. Available: https://github.com/Eclipse001/CroppingToolWithVTK

[5] P. S. Foundation. (2018, Apr.) 24.1. tkinter python interface to tcl/tk python 2.7.14 documentation. Python Software Foundation. [Online]. Available: https://docs.python.org/2/library/tkinter.html

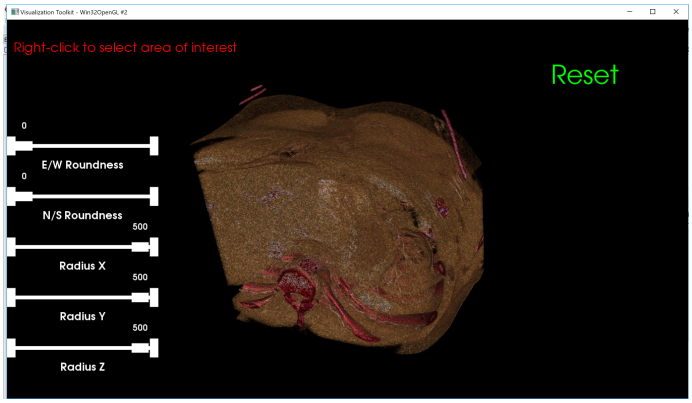

Fig. 3. Initial view of VTK windows.

Fig. 4. Cropping area centering the point selected.

Fig. 5. Adjust the shape of cropping area.

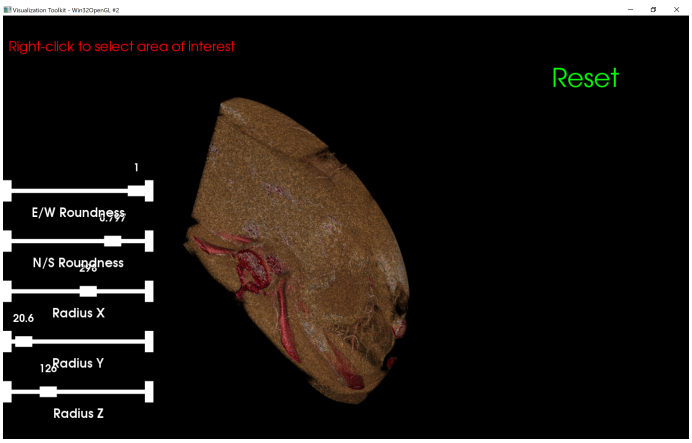

Fig. 6. Adjust the scale of cropping area.